

Date 27.04.2021

sivb@qti.qualcomm.com

Qualcomm

Machine Learning Training with TVM and Adreno GPU

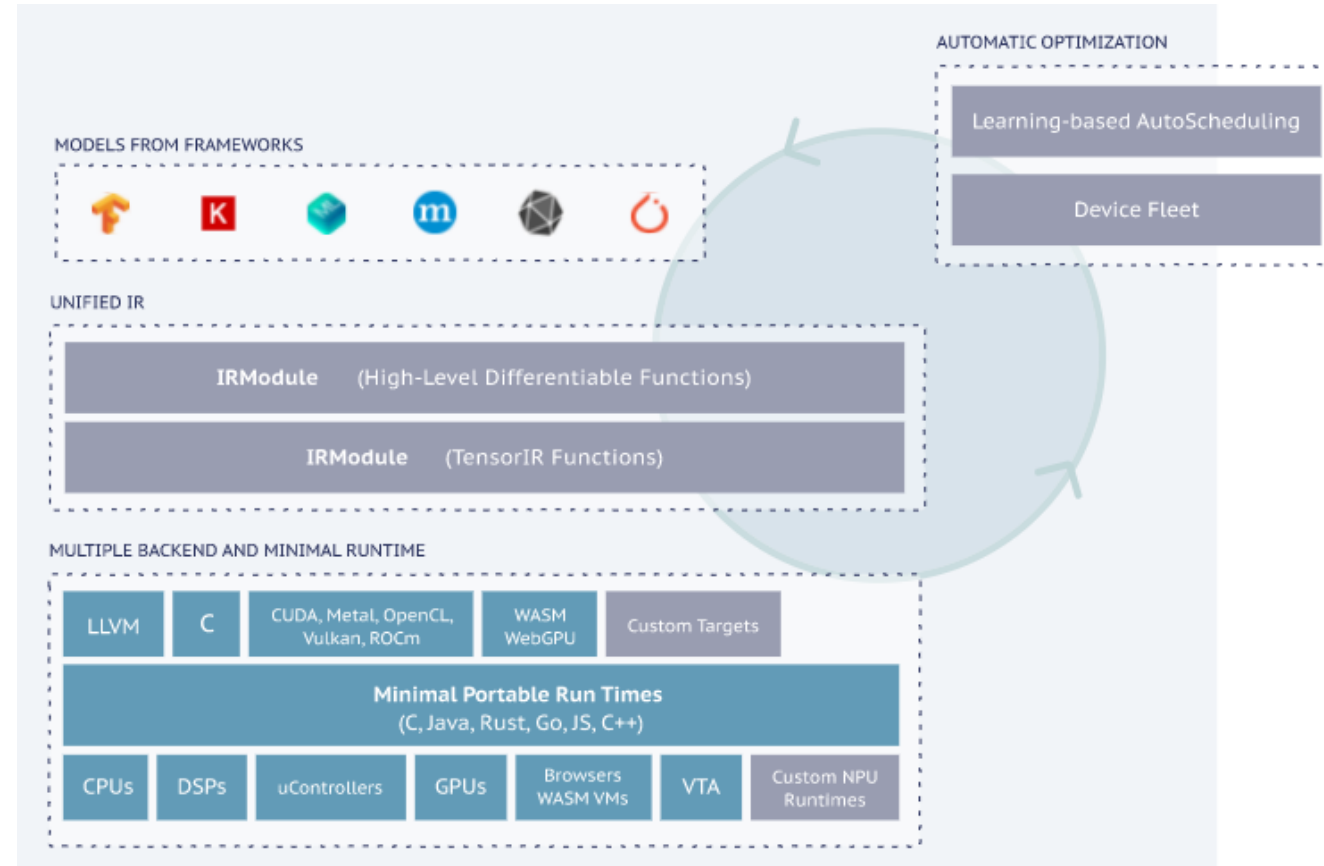
Siva



Introduction

TVM (Tensor Virtual Machine)

- An Apache project (<https://tvm.apache.org/>).
- Compiles deep learning models from various frames into minimum deployable modules.
- TVM Features
 - Frontend framework support.
 - Range of hardware.
 - Operating system and programming language compatibility.
 - AutoTVM
 - Auto Scheduler



<https://tvm.apache.org/>

In this work, we are presenting Neural network training on Qualcomm's Adreno mobile GPU using TVM framework. Current TVM doesn't demonstrate any sort of training. We improved TVM with necessary operators, graph transformations and workflow support required for DNN training. In this effort with OpenCL as target backend of TVM we could train LeNet-5 and MobilenetV1 from random parameter initialization.

Why TVM for Training

- **Minimal runtime:** Training naturally demands more resources compared to Inference. Optimized kernels and efficient use of memory resource is very important for mobile targets.
- **AutoTVM & AutoScheduler:** Can take full advantage of these for the training operations.
- **Auto Differentiation:** Decent support for AD for most of the operators.

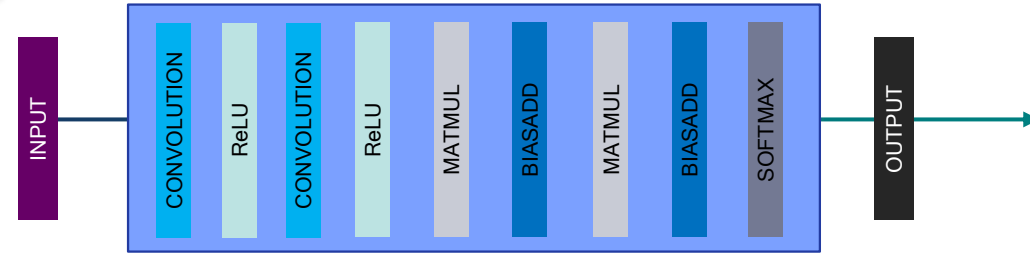
TVM's Limitation & Challenges

- Partial gradient operator support. No gradient support for operators like depthwise, batchnorm.
- TVM doesn't have any optimizers like SGD, Adam, RMSProp ...etc. which are essential for training.
- Gradient pass is partial. Existing gradient pass doesn't go well with MobilenetV1 to build training graph out of inference graph.
- Training Parameter handling: Every TVM iteration is independent and there exist no state information across iterations.
- Batchnorm: Differential behavior of Batchnorm for inference and training. Moving parameters handling.

* Limitations are w.r.t v0.7dev version of TVM.

LeNet-5

Forward Graph:



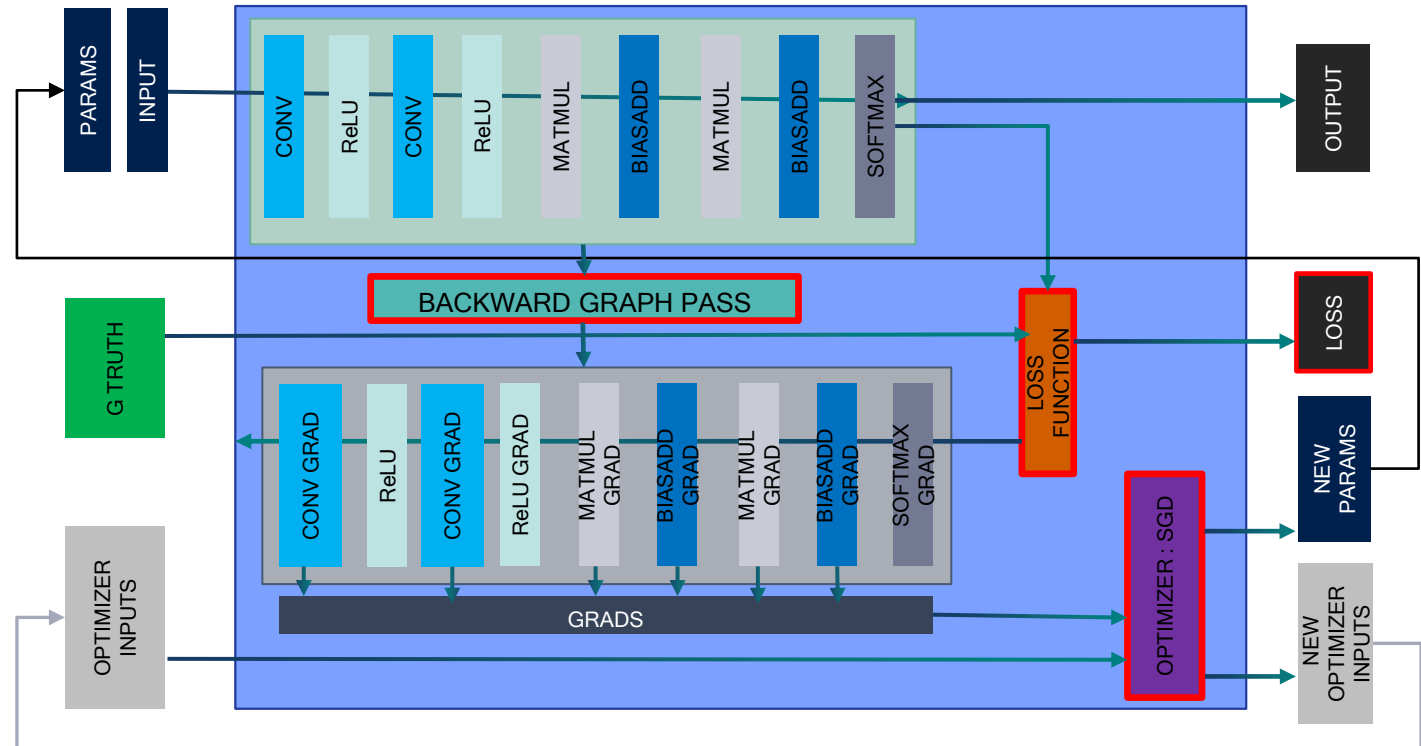
TVM Framework:

- Backward Graph Pass
- Gradients, Loss Functions
- Optimizers: SGD
- End to end Workflow

Adreno:

- Native App for Training

Backward Graph:

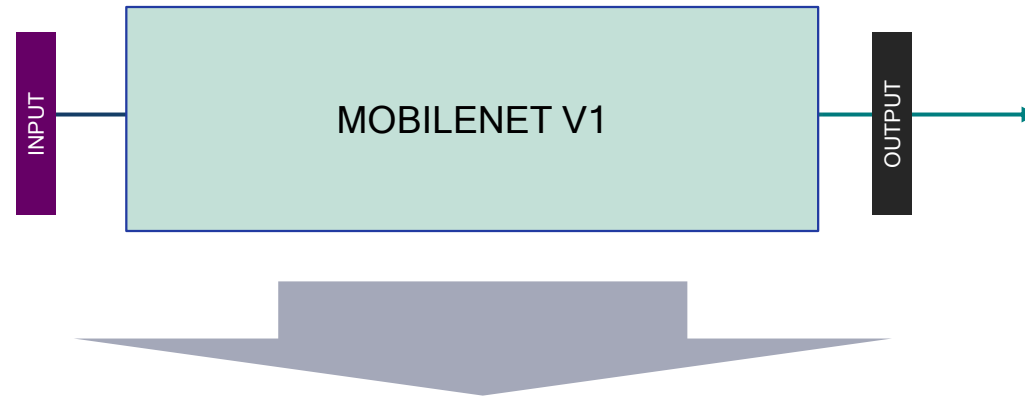


Mobilenet-v1

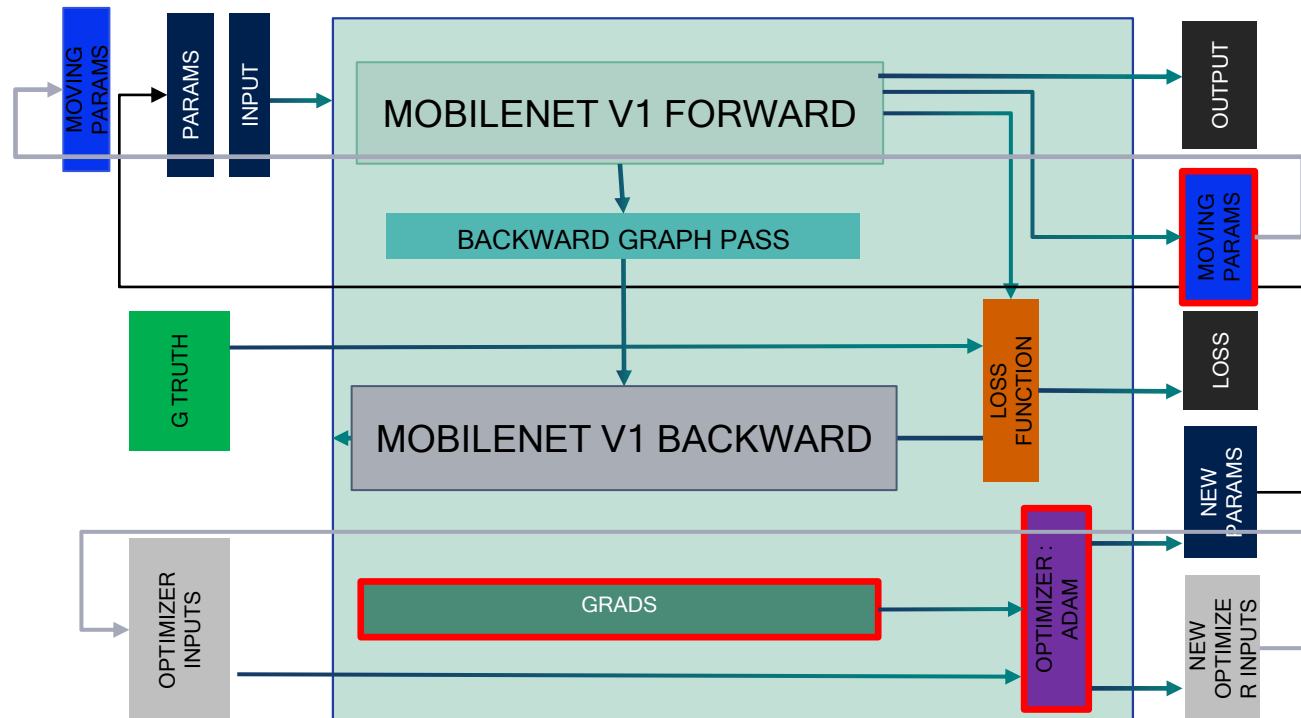
TVM Framework:

- More gradient ops : Depthwise & Batchnorm
- Optimizers: Adam, RMSProp
- End to end Workflow (Moving variables)
- Batchnorm: Differential behavior, moving param generation.

Forward Graph:



Backward Graph:



Summary of Experiments & Results



- **Network:** LeNet-5
 - **Dataset:** MNIST with 10K Samples with batch size = 32
 - **TVM:** Tensorflow Frontend, OpenCL Target, Tuning with AutoTVM.
 - **Result:** With “Uniform Random” initialization of initial params and SGD optimizer can hit ~98% in few epochs.
-
- **Network:** Mobilenet V1 with 128x128 input
 - **Dataset:** Cats & Dogs with 10K samples (80:20 split) with batch size = 2
 - **TVM:** Tensorflow Frontend, OpenCL Target, Tuning with AutoTVM.
 - **Result:** With “He” initialized initial params and Adam optimizer can hit ~94% in ~30 epochs.

Conclusions and Future work

- These efforts show case training capability of Adreno GPU using TVM framework.
- Backward graph builder pass we used here is limited to sequential graphs. This require enhancements to support branched network like Inception V3.
- Parameter handling across iterations does an additional copy which can be optimized with better memory planner in TVM.
- TVM compilation process can be improved to get a balance between memory and computational cost keeping in mind the resource limitations (specifically memory) of Target.
- Apart from end-to-end training, federated learning and transfer learning are potential solutions that can benefit here.
- Internal Adreno specific optimizations of TVM can be applied on training graphs with out any efforts.



Thank you!

Follow us on:   

For more information, visit us at:

www.qualcomm.com & www.qualcomm.com/blog

Nothing in these materials is an offer to sell any of the components or devices referenced herein.

©2018 Qualcomm Technologies, Inc. and/or its affiliated companies. All Rights Reserved.

Qualcomm is a trademark of Qualcomm Incorporated, registered in the United States and other countries. Other products and brand names may be trademarks or registered trademarks of their respective owners.

References in this presentation to “Qualcomm” may mean Qualcomm Incorporated, Qualcomm Technologies, Inc., and/or other subsidiaries or business units within the Qualcomm corporate structure, as applicable. Qualcomm Incorporated includes Qualcomm’s licensing business, QTL, and the vast majority of its patent portfolio. Qualcomm Technologies, Inc., a wholly-owned subsidiary of Qualcomm Incorporated, operates, along with its subsidiaries, substantially all of Qualcomm’s engineering, research and development functions, and substantially all of its product and services businesses, including its semiconductor business, QCT.